What's a little leakage between friends?

Sebastian Angel
University of Pennsylvania

David Lazar
MIT CSAIL

Ioanna Tzialla
New York University

Abstract

This paper introduces a new attack on recent messaging systems that protect communication metadata. The main observation is that if an adversary manages to compromise a user's friend, it can use this compromised friend to learn information about the user's *other* ongoing conversations. Specifically, the adversary learns whether a user is sending other messages or not, which opens the door to existing intersection and disclosure attacks. To formalize this *compromised friend attack*, we present an abstract scenario called the *exclusive call center problem* that captures the attack's root cause, and demonstrates that it is independent of the particular design or implementation of existing metadata-private messaging systems. We then introduce a new primitive called a *private answering machine* that can prevent the attack. Unfortunately, building a secure and efficient instance of this primitive under only computational hardness assumptions does not appear possible. Instead, we give a construction under the assumption that users can place a bound on their maximum number of friends and are okay leaking this information.

1 Introduction

In the past few years there has been a renaissance of messaging systems [2, 3, 17, 18, 26, 27] that allow users to communicate online without their messages being observed by ISPs, companies, or governments. These systems target a property called metadata-privacy which is stronger than end-to-end encryption: encryption hides the content of messages but it does not hide their existence nor any of their associated metadata (identity of the sender or recipient, frequency, time, and duration of communication, etc.). While hiding metadata has been the subject of a long line of work dating back three decades [7], there is renewed interest due to a proliferation of controversial surveillance practices [4, 6, 12–14], and the monetization of users' private information [5, 19, 21, 23].

Existing metadata-private messaging (MPM) systems guarantee that as long as the sender and the recipient of a message are not compromised, their communication cannot be observed by an adversary (the adversary learns that users are part of the system, but not whether they communicate). If either the sender or the recipient is compromised, MPM systems provide no guarantees (e.g., a compromised sender could trivially disclose to whom it is sending a message). In this paper we investigate whether an adversary—by

compromising and leveraging a user's friends—can learn anything about the user's *other* ongoing communications.

At first glance the answer to the above question appears to be no (assuming that the user does not voluntarily disclose the existence of other communications to compromised friends). After all, the guarantees of MPM systems should prevent the adversary from learning anything about conversations between uncompromised clients. Nevertheless, we find that this is not actually the case: engaging in a conversation with a compromised client consumes a limited resource, namely the number of concurrent conversations that a user can support. By observing a client's responses (or lack thereof), a compromised friend can learn whether the user has fully utilized this limited resource (i.e., the user is busy talking to others). In Section 5 we show how this one bit of information opens the door to existing intersection and disclosure attacks [1, 8–11, 15, 16, 20, 22, 24, 25], which invalidates the guarantees of MPM systems.

More interestingly, our *compromised friend attack* applies to all MPM systems that support a notion of *dialing* [18] (or any other mechanism that allows clients to start new conversations over time). We give a formal characterization of the attack with a scenario that we call the *exclusive call center problem*, which abstracts away the design or implementation of MPM systems. We then introduce a primitive called a *private answering machine* that solves the abstract problem and can be used by clients of MPM systems to prevent the compromised friend attack. In particular, clients use a private answering machine to select with which friends to communicate, while guaranteeing that compromised friends learn no information about other ongoing communications.

Unfortunately, building a cryptographically-secure private answering machine that does not require placing assumptions on the number of callers (i.e., the number of friends that a user can have) or incurring prohibitive delay or bandwidth appears hard. We compromise on this point and give a construction that can be used by MPM systems under the assumption that users can place a bound on their maximum number of friends. Our construction has two limitations: (1) it leaks the bound chosen by the user, and (2) it increases the latency of communication between a pair of users proportional to the chosen bound. Despite these limitations, our work addresses a previously overlooked attack vector and allows users in MPM systems to communicate online without leaking sensitive information.

In summary, the contributions of this work are:

- An abstract problem that captures the privacy leakage that results from oversubscribing a fixed resource in the presence of adversarial probing (§3).

- The *compromised friend attack* which exploits the fixed communication capacity of MPM systems (§5).

- The construction of a private answering machine (§4.1) that can be used in MPM systems to avoid leaking information to compromised friends about users' other ongoing conversations (§6).

2 Background

The goal of metadata-private messaging systems [2, 3, 17, 26, 27] is to allow a pair (or group) of *friends* to exchange bidirectional messages without leaking metadata to any party besides the sender and the recipient. A pair of users are friends if they have previously shared a secret, either out-of-band (e.g., in person at a coffee shop), or in-band with an *add friend* protocol [18]. Users in these systems exchange a fixed number of messages with their friends in discrete time epochs called rounds; users participate in every round even if they are idle. This ensures that an attacker that monitors the network cannot tell when users are actively communicating with their friends or starting/stopping conversations. This also places a bound on the number of active conversations that a user can have at any time; we refer to this bound as the client's *communication capacity*.

Once a client reaches its communication capacity, it cannot send messages to other friends until it ends an existing conversation. As a result, clients use a separate *dialing protocol* to coordinate the start and end of conversations. In a dialing protocol, a client sends a short message (a few bits) to a friend regardless of whether the friend's client has reached its communication capacity. The dialing message is sufficient to notify a user that one of their friends wishes to communicate, and to agree on a round to start the conversation [18]. There are multiple ways in which a client can react to a dialing message. Some natural choices are:

- If the client has not reached its communication capacity, it can automatically accept the call and start a new conversation.

- The client could prompt the user (similar to calling a friend in Skype), who can choose to accept or reject the call.

- If at capacity, the client could randomly end an existing conversation to make room for a new one.

Each of these choices is problematic. If the client's communication capacity is 1 (as in some of the existing systems [26, 27]) and the client automatically accepts calls, then any of the client's friends can easily learn when the client is *not* active in a conversation simply by calling. Leaving the choice to the users is slightly better since the user can choose to ignore or delay accepting some calls, but its choices can still inadvertently lead to intersection attacks. Ending conversations randomly hurts usability and might still leak information. The goal of the next section is to formalize the desired properties of the client's answering mechanism.

3 The exclusive call center problem

In order to avoid the details of particular MPM systems, we introduce an abstract scenario called the *exclusive call center problem*. It consists of a call center that has k operators capable of receiving calls (i.e., the call center has communication capacity k). The call center promises exclusivity to a single organization. This might be desirable to ensure high quality of service, for legal reasons, or to prevent the accidental leak of trade or business secrets to callers of a different organization. When a caller issues a call, an automatic *answering machine M* routes the call to an available operator who then processes the call. If M receives more calls than there are available operators, then M routes as many calls as it can, and notifies the remaining callers that all operators are busy.

While the above seems reasonable, the call center in question is greedy and wishes to oversubscribe its resources by contracting with a second organization—thereby violating its exclusivity agreement. This poses two problems for the call center. First, M cannot determine to which organization a call belongs; only an operator is in a position of making that distinction. Second, with the current decision logic of M (route to available operators, notify remaining callers that operators are busy), a group of callers from the same organization can collectively determine that they are not being given exclusive access to the call center (e.g., by placing k calls and noticing that not all are picked up). Given these issues and the limit of k operators (which is publicly known), can the call center do anything to maintain the illusion of exclusivity?

The first observation that the call center's CEO makes is that while there are k operators, there is no guarantee that all of them are available at any given point in time. After all, operators are human and take breaks. This, the CEO believes, opens the door to some level of plausible deniability. In particular, if M gives a caller from organization O_1 a busy signal it could mean:

(1) All k operators are busy handling other callers from O_1.

(2) Some operators are busy handling callers from O_1 and the remaining operators are on a break.

(3) Some operators are busy handling callers from O_1, some are busy handling callers from O_2, and some are on a break.

Possibility 1 is the expected scenario of a high-efficiency trustworthy call center. Possibility 2 is an unwanted outcome since it is inefficient, but it does not violate the contractual agreement. Possibility 3, however, violates the promise of exclusivity. The goal of the call center is to design M such that it is hard for either of the two organizations and their callers (assume no coordination between organizations) to infer that possibility 3 is the one taking place. As we alluded to earlier, the key challenge is that M cannot distinguish between callers (and determine to which organization they belong), and therefore cannot selectively lie to keep a consistent set of responses. We thus ask whether there exists an M that can leverage the proposed ambiguity to fool the organizations into thinking they are exclusive. In other words, is there a *private answering machine M*?

We think of M as acting in rounds, where in each round, M receives a set of calls C. We seek two informal properties from M.

- **Liveness**: eventually a caller in C gets to talk to an operator.

- **Privacy**: it is computationally hard for any colluding subset of callers $S \subseteq C$ (some of whom may get to speak to operators) to distinguish between a scenario where $S = C$ and a scenario where $S \subset C$ (i.e., it is difficult for the colluding subset of callers to determine whether they are the only callers or not).

The liveness guarantee is needed for M to be useful, but also to rule out a trivial solution: if M never puts anyone through to an operator, then the probability that any colluding set of callers S can distinguish between $S = C$ and $S \subset C$ is the same as flipping a coin.

Security game. To define privacy and liveness more formally, we use a security game played between an adversary $\mathcal{A}$ and a challenger parameterized by a polynomial time answering machine M and a security parameter λ. M takes as input a subset of callers C from the set of all possible callers $\mathbb{C}$, a communication capacity k, and a

random string r, where $k = poly(\lambda), |\mathbb{C}| = poly(\lambda), |r| = poly(\lambda)$. M outputs a set of callers $U \subseteq C$, such that $|U| \leq k$.

(1) $\mathcal{A}$ is given oracle access to M, and can issue a $poly(\lambda)$ number of queries to M with arbitrary inputs C, k, r. For each query, $\mathcal{A}$ can observe the corresponding result $U \leftarrow M(C, k, r)$.
(2) Challenger samples a random bit b uniformly in $\{0, 1\}$, and a random string r uniformly in $\{0, 1\}^\lambda$.
(3) $\mathcal{A}$ picks a set of callers S (where $S \subset \mathbb{C}$) and positive integer k, and sends them to the challenger.
(4) Challenger sets $C = S$ if $b = 0$, and $C = S \cup \{e\}$ if $b = 1$ (where e is a uniform random element from the set $\mathbb{C} - S$).
(5) Challenger calls $M(C, k, r)$ to obtain $U \subseteq C$ where $|U| \leq k$.
(6) Finally, the challenger removes e from U (if it is present) and returns the result $(U - \{e\})$ to $\mathcal{A}$.
(7) $\mathcal{A}$ outputs its guess b', and wins the security game if $b = b'$.

In summary, the adversary's goal in the game is to determine if the challenger is communicating with the uncompromised caller e after compromising all of the other callers (represented by S).

Definition 1 (Private answering machine). An answering machine M guarantees privacy if in the above security game with parameter λ, for all probabilistic polynomial time algorithms $\mathcal{A}$, there exists a *negligible function*[1] negl such that:

$$|\Pr[b = b'] - 1/2| \leq \mathrm{negl}(\lambda)$$

the probability is over the random coins of M and the challenger.

Definition 2 (Live answering machine). An answering machine M guarantees liveness if given security parameter λ, for any set of callers C, positive communication capacity k, and random string r, the probability that $M(C, k, r)$ outputs a non-empty set is non-negligible in λ. Here $|C| = poly(\lambda), k = poly(\lambda), |r| = poly(\lambda)$, and the probability is over the random coins of M.

4 Challenge with building answering machines

We discuss two straw man proposals to highlight the challenge of constructing an answering machine that meets Definitions 1 and 2.

Straw man M_1:

- **Input**: C, k, r
- $\pi \leftarrow$ uniform pseudorandom permutation of C according to r
- **Output**: the first $\min(k, |C|)$ elements from π

This is not secure. Let X be the random variable describing the cardinality of the set returned to $\mathcal{A}$, namely $|U - \{e\}|$. Assuming that $k \leq |C|$, $\Pr[X < k \mid b = 0] = 0$ and $\Pr[X < k \mid b = 1] = k/|C|$. As a result, $\mathcal{A}$ can, by simply counting the elements in $U - \{e\}$, distinguish between $b = 0$ and $b = 1$ with non-negligible advantage.

Straw man M_2:

- **Input**: C, k, r
- $\pi \leftarrow$ uniform pseudorandom permutation of C according to r
- Sample $m \in [0, \min(k, |C|)]$ uniformly at random
- **Output**: the first m elements from π

[1]A function $f : \mathbb{N} \to \mathbb{R}$ is negligible if there exists an integer c such that for all positive polynomials $poly$ and all x greater than c, $|f(x)| < 1/poly(x)$.

This is also not secure. Let $k = 1$ and $|S| = 1$. The probability that the challenger returns to $\mathcal{A}$ an empty set is higher when $b = 1$ due to Line 6 in the security game and the way we construct M_2. Again, let X be the random variable describing the cardinality of the set returned to $\mathcal{A}$. In particular, $\Pr[X < 1 \mid b = 0] = 1/2$, whereas $\Pr[X < 1 \mid b = 1] = 3/4$. As a result, $\mathcal{A}$ can distinguish between $b = 0$ and $b = 1$ with non-negligible advantage. More generally, since X is drawn from a uniform distribution when $b = 0$, the probability mass function (pmf) for X (assuming $k \leq |C|$) is:

$$f(x) = \begin{cases} \frac{1}{k+1} & \text{for } 0 \leq x \leq k \\ 0 & \text{otherwise} \end{cases}$$

On the other hand, if $b = 1$, the pmf for X is:

$$f(x) = \begin{cases} \frac{1}{k+1} + \frac{1}{2(k+1)} & \text{for } x = 0 \\ \frac{1}{k+1} & \text{for } 1 \leq x \leq k - 1 \\ \frac{1}{2(k+1)} & \text{for } x = k \\ 0 & \text{otherwise} \end{cases}$$

An adversary $\mathcal{A}$ can leverage the difference in these pmfs to distinguish between $b = 0$ and $b = 1$ with non-negligible advantage.

We could sample m and permute C non-uniformly, but the effect of Line 6 is large enough for $\mathcal{A}$'s advantage to remain non-negligible (M must output a non-empty set with non-negligible probability to satisfy liveness). As a result, building an M that guarantees privacy and liveness when $k < |\mathbb{C}|$ seems hard. Below we give the construction of a private answering machine under a relaxed setting.

4.1 Answering machine with a known set of callers

We now discuss the construction of an answering machine that provides privacy and liveness under the assumption that the set of all possible callers ($\mathbb{C}$) is fixed and known in advance to M (the machine still does not know which callers belong to a particular organization). As a result, we assume that each element e in $\mathbb{C}$ can be uniquely mapped to an integer in the range $[1, |\mathbb{C}|]$ with the function $id(e)$, and that this mapping is known to M.

Private and live answering machine M_3:

- **Input**: C, k, r
- $U \leftarrow \varnothing$
- $\forall_{e \in C}, \forall_{0 \leq i < k}$, if $id(e) \equiv (r + i) \mod |\mathbb{C}|$, add e to U
- **Output**: U

In other words, M_3 precomputes a schedule mapping callers to rounds: in each round a set of k callers will be serviced (the input r is the current round). If a caller happens to call during a round that has been allocated for it, it will be added to the set U (i.e., its call will be handled). Otherwise, the call will not be answered.

Machine M_3 guarantees liveness because for every caller e, every k out of $|\mathbb{C}|$ rounds are assigned to e; since $|\mathbb{C}| = poly(\lambda)$, this occurs with non-negligible probability. Machine M_3 guarantees privacy because the response given to $\mathcal{A}$ at the end of the game (Step 6 in the security game) depends only on r and not on b. As a result this response is exactly the same when $b = 0$ and $b = 1$; observing this response gives no advantage to $\mathcal{A}$.

5 The compromised friend attack on MPM systems

The exclusive call center problem is the scenario encountered by users in MPM systems who communicate with compromised friends. Clients in these systems can only handle a fixed number of concurrent conversations in one round (this maps to the k operators in the call center problem), which opens the door to an attack that we call the *compromised friend* (CF) attack. An adversary—via compromised friends—can dial (or start a conversation through any other means supported by the MPM system) a client and observe whether the client responds or not. If the client does not have a private answering machine, the adversary can distinguish between a scenario where the client is talking to some honest client (i.e., the adversary's subset of callers is not the full set, $S \subset C$), and a scenario where the client is not ($S = C$). This can leak one bit of information that opens the door to existing attacks.

Intersection, disclosure, and hitting set attacks. There is a large literature of traffic analysis attacks that uncover patterns of communication by observing when users send and receive messages. For example, intersection attacks [24] can be used to narrow down the possible recipients of a message when users communicate with a single friend, while disclosure [1] and hitting set [16] attacks can handle the case where users communicate with multiple friends. There are also statistical variants of these attacks [8].

MPM systems purportedly avoid these attacks by requiring clients to always be online, continuously sending and retrieving messages; the client sends dummy requests if the user is idle. Unfortunately, the CF attack allows an adversary to guess whether a client is sending dummy messages or not with non-negligible advantage. An adversary can therefore target a set of potential senders and recipients with a CF attack, making these systems vulnerable to traffic analysis. Note that the CF attack can be achieved in another way: If a pair of friends is currently communicating at a rate of r messages per round ($r < k$), and they wish to increase this rate to improve their throughput, this is the moral equivalent of dialing (since it consumes a client's limited communication capacity).

Difficulty of conducting a CF attack in practice. There are some challenges in composing a CF attack with existing attacks. First, depending on the answering mechanism of the MPM system, an adversary might need to conduct a CF attack many times before learning anything useful (recall from Section 3 that while the adversary has non-negligible advantage, it might still be small). However, existing MPM systems (e.g., [3, 18, 27]) currently implement an answering mechanism that leaks information with a single CF attack. Second, the CF attack requires actively targeting users on a given round, which may limit the number of observations that are available to an adversary. Last, this attack requires compromising users' actual friends or it requires the use of phishing attacks to fool users into befriending malicious users.

6 Mitigation strategies for MPM systems

We now discuss ways in which MPM systems could prevent a CF attack. One option is for clients to use a private answering machine such as M_3 (§4.1) to determine which of the new (or existing) conversations to accept (or continue) without leaking information. Clients would continue to exchange k messages per round, but only a subset of these messages (based on the output of M_3) would correspond to actual conversations; the rest would act as cover traffic. Note that with M_3, a compromised friend can learn how many other friends a user has, or at least an upper bound on it (i.e., $|\mathbb{C}|$). Furthermore, M_3 accepts messages from a particular friend for a sliding window of k rounds, so it is possible for two users' sliding windows to never intercept. As a result, if the MPM system does not allow the retrieval of messages from previous rounds, clients would be unable to communication without additional mechanisms.

In principle, when using M_3, we could set $\mathbb{C}$ to be all users in the system rather than just a client's friends (so the adversary learns no information); M_3's function *id* could be computed with a collision-resistant hash function. Provided that the number of total users (n) is $poly(\lambda)$, this would technically meet our liveness requirement (Definition 2), In practice, however, this would result in a client accepting a call from a given friend every k out of n rounds, which is a prohibitive delay when n is large.

The alternative to using a private answering machine is for clients to set their communication capacity (k) to a value larger than their maximum number of friends (under the assumption that each pair of friends exchanges at most one message per round). This too would leak the bound on the number of friends of a given client. If a client wishes to keep this information private, a client could set k to be the total number of users in the system. While this would leak no information, the communication and computational costs of existing MPM systems increase linearly with k (though some systems have sublinear computational costs [3]), making it prohibitive for systems with many users. More worryingly, several MPM systems [26, 27] provide guarantees that are based on differential privacy, and increasing the number of concurrent conversations (k) accelerates the consumption of users' privacy budgets.

7 Summary and discussion

In this work we introduced an attack that allows an adversary to violate the guarantees of MPM systems by leveraging users' friends. We also proposed several mitigations, but our proposals satisfy only two out of three desirable properties: privacy (leak no information), low communication overhead (i.e., clients need not send many messages per round), and low latency (friends get to talk to each other often). The most pragmatic of our solutions requires bounding the maximum number of friends that a client can have.

Even with our mitigations, compromised friends are a liability and can be used to learn sensitive information through other means. For example, if a user is uncharacteristically slow to respond to a compromised friend's message (a user's response pattern could be constructed over many prior interactions), this anomaly in itself could leak information. We believe that understanding the practical impact of this type of attack is a promising avenue for future work.

References

[1] D. Agrawal and D. Kesdogan. Measuring anonymity: The disclosure attack. *IEEE Security & Privacy*, 1(6), Nov. 2003.

[2] N. Alexopoulos, A. Kiayias, R. Talviste, and T. Zacharias. MCMix: Anonymous messaging via secure multiparty computation. In *Proceedings of the USENIX Security Symposium*, Aug. 2017.

[3] S. Angel and S. Setty. Unobservable communication over fully untrusted infrastructure. In *Proceedings of the USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, Nov. 2016.

[4] J. Ball. GCHQ captured emails of journalists from top international media. http://goo.gl/YzXnYK, Jan. 2015. The Guardian.

[5] J. Bamford. Shady companies with ties to Israel wiretap the U.S. for the NSA. http://goo.gl/bdi7w4, Apr. 2012. Wired.

[6] S. Buttar. Dragnet NSA spying survives: 2015 in review. https://goo.gl/JsNgS7, Dec. 2015. Electronic Frontier Foundantion.

[7] D. L. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2), Feb. 1981.

[8] G. Danezis. Statistical disclosure attacks. In *Proceedings of the IFIP Information Security Conference*, May 2003.

[9] G. Danezis, C. Diaz, and C. Troncoso. Two-sided statistical disclosure attack. In *Proceedings of the Workshop on Privacy Enhancing Technologies (PET)*, June 2007.

[10] G. Danezis and A. Serjantov. Statistical disclosure or intersection attacks on anonymity systems. In *Proceedings of the International Workshop on Information Hiding*, May 2004.

[11] G. Danezis and C. Troncoso. Vida: How to use bayesian inference to de-anonymize persistent communications. In *Proceedings of the Privacy Enhancing Technologies Symposium (PETS)*, Aug. 2009.

[12] R. Dingledine. Did the FBI pay a university to attack Tor users? https://goo.gl/NB3hSR, Nov. 2015. Tor Project.

[13] G. Greenwald and R. Gallagher. New Zealand launched mass surveillance project while publicly denying it. https://goo.gl/UwNpwV, Sept. 2014. The Intercept.

[14] G. Greenwald and E. MacAskill. NSA Prism program taps in to user data of Apple, Google and others. http://goo.gl/qETWUq, June 2013. The Guardian.

[15] D. Kesdogan, D. Mölle, S. Ritchter, and P. Rossmanith. Breaking anonymity by learning a unique minimum hitting set. In *Proceedings of the International Computer Science Symposium in Russia (CSR)*, Aug. 2009.

[16] D. Kesdogan and L. Pimenidis. The hitting set attack on anonymity protocols. In *Proceedings of the International Workshop on Information Hiding*, May 2004.

[17] A. Kwon, H. Corrigan-Gibbs, S. Devadas, and B. Ford. Atom: Horizontally scaling strong anonymity. In *Proceedings of the ACM Symposium on Operating Systems Principles (SOSP)*, Oct. 2017.

[18] D. Lazar and N. Zeldovich. Alpenhorn: Bootstrapping secure communication without leaking metadata. In *Proceedings of the USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, Nov. 2016.

[19] R. Lenzner. ATT, Verizon, Sprint are paid cash by NSA for your private communications. http://goo.gl/x7Cz1m, Sept. 2013. Forbes.

[20] N. Mallesh and M. Wright. The reverse statistical disclosure attack. In *Proceedings of the International Workshop on Information Hiding*, June 2010.

[21] J. Menn. Yahoo secretly scanned customer emails for U.S. intelligence. https://goo.gl/KZuUYo, Oct. 2016. Reuters.

[22] F. Pérez-González and C. Troncoso. Understanding statistical disclosure: A least squares approach. In *Proceedings of the Privacy Enhancing Technologies Symposium (PETS)*, July 2012.

[23] E. Protalinski. Facebook scans chats and posts for criminal activity. http://goo.gl/pfV9XE, July 2012. CNET.

[24] J.-F. Raymond. Traffic analasis: Protocols, attacks, design issues, and open problems. In *Proceedings of the International Workshop on Design Issues in Anonymity and Unobservability*, July 2000.

[25] C. Troncoso, B. Gierlichs, B. Preneel, and I. Verbauwhede. Perfect matching disclosure attack. In *Proceedings of the Privacy Enhancing Technologies Symposium (PETS)*, July 2008.

[26] N. Tyagi, Y. Gilad, D. Leung, M. Zaharia, and N. Zeldovich. Stadium: A distributed metadata-private messaging system. In *Proceedings of the ACM Symposium on Operating Systems Principles (SOSP)*, Nov. 2017.

[27] J. van den Hooff, D. Lazar, M. Zaharia, and N. Zeldovich. Vuvuzela: Scalable private messaging resistant to traffic analysis. In *Proceedings of the ACM Symposium on Operating Systems Principles (SOSP)*, Oct. 2015.